RECEIVED

JUL 22 1996

OSTI

Interactive Graphical Tools for Three-Dimensional Mesh Redistribution

Lauri Ann Dobbs
(MS Thesis)

March 1996

Lawrence
Livermore
National
Laboratory

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

This report has been reproduced
directly from the best available copy.

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information
P.O. Box 62, Oak Ridge, TN 37831
Prices available from (615) 576-8401, FTS 626-8401

Available to the public from the
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd.,
Springfield, VA 22161

Interactive Graphical Tools for Three-Dimensional Mesh Redistribution

**Lauri Ann Dobbs
(MS Thesis)**

Manuscript date: March 1996

LAWRENCE LIVERMORE NATIONAL LABORATORY
University of California • Livermore, California • 94551

Interactive Graphical Tools
for
Three-Dimensional Mesh Redistribution

By

Lauri Ann Dobbs

B.S. (California Polytechnic State University, San Luis Obispo) 1984

THESIS

Submitted in partial satisfaction of the requirements for the degree of

MASTER OF SCIENCE

in

Computer Science

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Mesa M. Blüthner
Rebecca R. Sprünger
Nelson L. Max

Committee in Charge

1996

Acknowledgments

I would like to express my gratitude to my thesis advisors: Dr. Meera Blattner, Dr. Rebecca Springmeyer, and Dr. Nelson Max. I appreciate the time they spent with me and their support and guidance. Thanks to Brad Maker formerly of the Methods Development Group at Lawrence Livermore National Laboratory for his ideas on the topic for this thesis. I also want to thank the scientists that willingly participated in the observation and interviews as well as the prototype tool evaluation. Without their time and ideas this thesis could not have been completed. I also would like to express my gratitude to Dr. Woodruff Clark the instructor for my last class: Human-Computing Interaction. His class provided a forum for discussing and formulating many of the ideas in this thesis concerning the study and analysis of processes.

I especially want to thank Evi Dube, a special friend and fellow student. She participated in the observation sessions; lent reference materials; provided moral support, ideas, and helped clarify numerous issues; and above all listened.

A very special thanks to my husband Dave and children Jamie and Crystie for their support, patience, and understanding. We all spent many hours side by side "doing our homework (which the girls think is synonymous with thesis)". Thanks to my parents, sisters, and their families for their support and for taking the girls on weekends while I studied.

Lauri Ann Dobbs
March 1996
Computer Science

Interactive Graphical Tools
for
Three-Dimensional Mesh Redistribution

Abstract

Three-dimensional meshes modeling nonlinear problems such as sheet metal forming, metal forging, heat transfer during welding, the propagation of microwaves through gases, and automobile crashes require highly refined meshes in local areas to accurately represent areas of high curvature, stress, and strain. These locally refined areas develop late in the simulation and/or move during the course of the simulation, thus making it difficult to predict their exact location. This thesis is a systematic study of new tools scientists can use with redistribution algorithms to enhance the solution results and reduce the time to build, solve, and analyze nonlinear finite element problems. Participatory design techniques including Contextual Inquiry and Design were used to study and analyze the process of solving such problems. This study and analysis led to the in-depth understanding of the types of interactions performed by FEM scientists. Based on this understanding, a prototype tool was designed to support these interactions. Scientists participated in evaluating the design as well as the implementation of the prototype tool. The study, analysis, prototype tool design, and the results of the evaluation of the prototype tool are described in this thesis.

Table of Contents

TABLE OF ILLUSTRATIONS.....	6
CHAPTER 1	7
INTRODUCTION.....	7
1.1 Overview.....	7
1.2 Organization.....	10
CHAPTER 2	12
RELATED TOPICS.....	12
2.1 Study and Analysis of Processes.....	12
2.2 Current Mesh Tools.....	20
2.2.1 Mesh Generator.....	21
2.2.2 Unstructured Two-Dimensional Rezoner	24
2.2.3 Structured Rezoner	27
CHAPTER 3	29
OBSERVATIONS AND ANALYSIS	29
3.1 Interviews and Observations	29
3.2 Analyzing the Data	33
CHAPTER 4	42
AN INTERACTIVE 3D REZONER.....	42
4.1 Design.....	42
4.1.1 Controller	44
4.1.2 Solver.....	44
4.1.3 Post-processor	44
4.1.4 Rezoner.....	45
4.2 Prototype Tool.....	46
CHAPTER 5	54
REZONER EVALUATION.....	54
5.1 Evaluation Process.....	54
5.2 Design Evaluation Results.....	56
5.3 Prototype Tool Evaluation Results.....	58
CHAPTER 6	64
DISCUSSION.....	64
6.1 Advantages and Disadvantages of the Approach.....	65
6.2 Future Work.....	69
GLOSSARY	71
REFERENCES	74
APPENDIX A.....	78
INTERVIEW QUESTIONS.....	78
APPENDIX B.....	79
QUALITY MEASURE CALCULATIONS.....	79

APPENDIX C.....	91
COMMAND SUMMARY.....	91

Table of Illustrations

FIGURE 1-1. A MESH FOR A SUPERPLASTIC FORMING ANALYSIS.	8
FIGURE 2-1. PROCESS-ORIENTED METHODOLOGY [SPRINGMEYER, 92].	16
TABLE 2-1. NGP MESH QUALITY MEASURES.	23
TABLE 2-2. A SAMPLE COMMAND SEQUENCE FOR NIKE2D REZONING.	26
TABLE 3-1. SUMMARY OF INTERVIEWS.	30
FIGURE 3-1. SUMMARY OF AN OBSERVATION SESSION.	34
FIGURE 3-2. DECOMPOSING THE PROCESS INTO CATEGORIES OF RELATED ACTIONS.	35
TABLE 3-2. ACTIONS ASSOCIATED WITH EACH SUBCATEGORY.	37
FIGURE 3-3. PHYSICAL MODEL FOR SOLVING FINITE ELEMENT PROBLEMS.	38
FIGURE 3-4. SEQUENCE MODEL FOR THREE-DIMENSIONAL FINITE ELEMENT PROBLEMS.	39
FIGURE 3-5. SEQUENCE MODEL FOR TWO-DIMENSIONAL FINITE ELEMENT PROBLEMS.	40
FIGURE 4-1. THREE-DIMENSIONAL REZONER DESIGN.	43
FIGURE 4-2. SKEWNESS OF THREE-DIMENSIONAL MODEL.	48
TABLE 4-2. PEELING COMMANDS.	49
FIGURE 4-3. PEELING WITH SKEWNESS.	51
FIGURE 4-4. QUALITY QUERY MENU AND RESULTS.	53
FIGURE 5-1. PROTOTYPE/TEST DRIVE CYCLE [DUNCAN & BEABES, 95].	55
FIGURE 5-2. MODIFIED THREE-DIMENSIONAL REZONER DESIGN.	57
FIGURE 5-3. TIME HISTORY PLOT OF ASPECT RATIO.	60
FIGURE 5-4. QUALITY MENU REDESIGN.	62
FIGURE G-1. DATA FLOW DIAGRAM [DE MARCO, 78].	71
FIGURE G-2. EXAMPLE OF INFORMATION STRUCTURE DIAGRAM [SHLAER & MELLOR, 88].	72

Chapter 1

Introduction

1.1 Overview

Finite Element Methods (FEM) appear in many scientific and technical fields, including fluid mechanics, aerodynamics, biomechanics, geomechanics, thermal analysis, acoustic, electric and magnetic field modeling, shock and impact modeling, reactor physics, and plasma modeling [Thalman, 90]. Usually three steps requiring at least three separate programs are required to solve finite element problems:

1. Represent the problem as a mesh (an array of grid points labeled and interconnecting in some way to define a discretization of the domain as a union of mesh cells or elements [Carey, 92]) using an interactive, graphical mesh generator.
2. Find a solution to the problem using a non-interactive numerical solver such as DYNA3D [Whirley & Engelmann, 93] or NIKE3D [Maker *et al.*, 91].
3. Interpret the solution using an interactive, graphical post-processor.

Some problems require highly refined meshes in local areas to accurately model areas of high curvature or concentrated strain, for example. For nonlinear problems¹ such as metal forming, forging, and crashworthiness studies solved with Lagrangian solvers, the areas in need of local refinement develop late in the simulation and/or move during the course of the simulation. With the tools currently available, scientists must predict where to

¹According to [Goudreau, 93] *linearity* is a relation between two quantities (for example, stress and strain) when a change in one produces a directly proportional change in the other. *Nonlinearity* means the relationship between quantities is complex and not directly proportional. A car bumper hitting a barrier is an example of nonlinear finite element problem.

refine the original mesh to ensure that the refined areas coincide with the areas of interest throughout the simulation. After reviewing the results of the simulation, scientists will modify the initial mesh using a mesh generator to refine areas of the mesh, rerun the finite element solver, and reexamine the solution several times before an accurate solution is generated. A superplastic forming analysis [Maker, 93] displayed in Figure 1-1 demonstrates the need for locally refined areas. The diagram to the left is a mesh of one quarter of a piece of sheet metal. Pressure is applied to the initially flat sheet metal blank to form a pan. Eight iterations were needed to build this mesh where the locally refined areas form the corners of the pan at the end of the analysis. The diagram to the right is the results of the simulation: one quarter of the pan. Note that the refined areas of the mesh line up with the corners of the pan. The contours represent the thickness of the sheet metal.

Figure 1-1. A mesh for a superplastic forming analysis.

Redistribution and refinement are two approaches developed to simplify this processes of local refinement of a mesh. Refinement adds or deletes nodes from the mesh in areas of interest. Redistribution moves existing nodes in order to cluster nodes in areas of interest. Redistribution is also referred to as grid optimization or rezoning. Carey [92] outlines various automated refinement and redistribution algorithms. These algorithms, integrated into the solvers, use error estimators to detect when and where refinement is necessary, and then automatically refine those areas of the mesh. This automated approach works well for linear problems because error estimators can be defined to determine when and where the mesh needs refining. However, it is difficult to define error estimators for nonlinear problems because the relationships between quantities such as stress and strain are more complex. Thus for complex nonlinear problems, it is more effective for scientists to interactively redistribute nodes using three-dimensional graphical tools. These tools give the scientists precise control over where and when the mesh should be refined. Interactive tools have been developed for two-dimensional structured and unstructured meshes and three-dimensional structured meshes [Engelmann, 91; Lu & Eiseman, 91]. Interactive rezoners for unstructured three-dimensional meshes have not been developed because it is difficult to develop easy-to-use and efficient interfaces for manipulating and visualizing three-dimensional objects. However, with the continuous advancements in desk-top computers, scientists are now analyzing more complex three-dimensional problems increasing the demand for interactive rezoning tools.

This thesis is a systematic study of new tools scientists can use with redistribution algorithms to enhance the solution results and reduce the time to build, solve, and analyze

three-dimensional nonlinear finite element problems. Qualitative methods are used to study the scientists' actions in solving nonlinear finite element problems, so that software developers can better understand those actions and design tools to support them. Contextual Inquiry and Design are qualitative methods used in this thesis to develop an understanding of the process of solving nonlinear finite element problems, analyze scientists' actions, evaluate tools currently used to support those actions, and to analyze the sequence (*i.e.*, ordering) of those actions. This thesis describes the qualitative methods used to study the process of solving nonlinear finite element problems, the results of the study, how those results led to the development of the prototype tool, the prototype tool, and an evaluation of the prototype tool.

1.2 Organization

This thesis is organized into six chapters. The first chapter presents an overview of work related to this thesis. It covers qualitative methods for studying processes and for evaluating prototype tools and new tools. This chapter includes a description of the mesh quality evaluation capabilities found in a state-of-the-art mesh generator and the redistribution capabilities found in a two-dimensional unstructured mesh rezoner and a structured three-dimensional mesh rezoner.

Chapter 3 describes three techniques used to study and analyze the process of solving nonlinear finite element problems: Contextual Inquiry, Contextual Design, and a process-oriented approach [Springmeyer, 92]. The study in this thesis identifies the actions

scientists take to solve these problems, the sequence of these actions, the tools supporting these actions (including those not on computers), and the effectiveness of these tools.

From the results of the study a design for a three-dimensional rezoner and a prototype tool were developed. The prototype tool supports the scientists' actions of interactively examining a mesh and identifying areas for local refinement. The prototype tool supports only a portion of the actions identified as necessary for three-dimensional rezoning because the development of a fully functional rezoner is beyond the scope of this thesis. Chapter 4 describes the rezoner design and the prototype tool.

Contextual Inquiry methods were used to evaluate the usefulness of the prototype tool. The process for evaluating the prototype tool and results of the evaluation are described in Chapter 5.

The final chapter discusses the advantages and disadvantages of this approach to designing computer-based tools and considers possible future extensions to this work.

Chapter 2

Related Topics

The following sections discuss work related to this thesis. Section 2.1 describes problems with traditional software analysis approaches and suggests alternative techniques. These techniques include Contextual Inquiry, Contextual Design, and Springmeyer's process-oriented approach. Section 2.2 describes interactive tools that scientists currently use to build and fix meshes for finite element analysis. These tools include a state-of-the-art mesh generator and interactive rezoning tools for two-dimensional unstructured meshes and structured three-dimensional meshes.

2.1 Study and Analysis of Processes

The goal of a software designer is to build software systems that are delivered on-time and within budget; that support, extend, and transform the user's work; and that are easy to maintain, modify, and understand. To build such systems designers must understand the user's work and the tools currently in use. In many cases, designers have little knowledge of that work or the context within which their tools will be used. The question is "how do designers with little knowledge of the user's work quickly learn enough about the work in order to build a useful system?"

Analysis methodologies such as Structured Analysis [De Marco, 78] and Object Oriented Analysis [Shlaer & Mellor, 88] have attempted to answer this question. These analysis methodologies focus on helping the designer identify, formulate, and verify experts' knowledge of the problem domain. With the use of surveys, traditional interviews, and the evaluation of current tools a designer starts by building a conceptual model of the current environment in either the form of a data flow diagram or an information model. The conceptual model is a graphical means for defining and describing the users' vocabulary and identifying, classifying, and abstracting the problem domain. With training this model can be shared with users and other members of the design and implementation team in order to verify the accuracy of the model and formulate a common view of the problem domain. A new model is then formulated by either modifying the current conceptual model or building an entirely new model to reflect the incorporation of the new software system into the users' environment. This model is a "paper model of the system to be installed" [De Marco, 78] and is shared with the users to verify that the new system will meet their needs prior to it being built. With these methodologies, the users provide the data to build the conceptual model and review them for accuracy and feasibility early in the software development cycle, thereby increasing the probability that a successful tool will emerge from this process. However, the techniques suggested for gathering data are insufficient for gathering information about the user's interactions with the system and their intent. Insufficient data about the users' interactions and intents generally lead to the development of a system that does not support the users' work.

In traditional interviews, interviewers direct their subject's conversation by asking questions and supplying the topics for discussion. This limits the information collected to that data that answers the interviewers questions. In the case where the software designer is the interviewer, the interviewer's lack of knowledge in the area under investigation limits his/her ability to generate questions that will provide an accurate understanding of the process. In addition, the subject's responses may be inaccurate. According to [Ericsson & Simon, 93] in an interview the interviewer asks a subject to recall information about a process that is completed and cannot be altered or influenced. If asked to report information that was never heeded, the subject is likely to "infer and generate an answer." This is typically the case with describing a procedure or process and with "why" questions such as "why did you do ..?" While reporting on a familiar procedure, the subject (*e.g.*, a scientist analyzing his/her data or a mechanical engineering solving a finite element problem) might report: 1) the general procedures he/she applied without reference to any specific behavior; 2) a procedure he/she formulated based on the memory of some parts of a specific procedure; 3) how the procedure ought to be based on prior information the subject has on the process; and/or 4) a general procedure omitting some recurring intermediate steps because these steps or actions become automatic and the subject becomes unconscious of his/her actions². In all cases of reporting procedures,

²[Bodker, 89] calls these unconscious actions "operations". Operations are bound to specific material conditions. The right operation is triggered by the material conditions. An example she uses is in the composition of a newspaper page, in placing an ad in the lower right-hand corner an artist will use operations such as holding a knife, cutting the paper, and trying to position the ad on the paper. Only when there is a breakdown, something doesn't work right, will the subject become conscious of the operations (*i.e.*, the knife breaks or can't be found).

traditional interviews provide insufficient detail for designing new tools to support users' work.

Surveys have been the leading technique for gathering data in the area of social science for many years and are generally used to collect general objective information about the subject under study (*e.g.*, range of income, age group of household occupants). For designing software systems, surveys may be an effective technique to gather similarly objective information such as the functionality of existing tools. However, information about a user's work and his/her intent is subjective. Questions with a fixed set of alternative answers (*e.g.*, yes/no or multiple choice) typically found in surveys force the participant to "infer" an answer if the appropriate answer is not available or the participant never heeded the information the question is asking to report. Essay questions such as "Explain how you build a mesh" would require recalling information after the fact, and therefore have similar problems to those mentioned about traditional interviews. Surveys have the additional limitation that the data collected is limited to those that participate which may not be a representative sample.

Springmeyer [92] recognized the deficiencies with traditional approaches and developed a process-oriented methodology for observing and characterizing how scientists analyze their data. Her process-oriented methodology focuses on the process to be supported rather than the users, tools, or the design of one particular tool. This methodology provides a means for studying a process that involves an individual in a complex activity unfamiliar to the software developer. Her methodology as depicted in Figure 2-1 below

combines aspects of Contextual Inquiry [Wixon & Holtzblatt & Knox, 90], participatory observation [Warwick & Lininger, 75], and interaction analysis [Tang, 89; Minneman, 91] to discern the user's intent as well as his/her actions.

Figure 2-1. Process-Oriented Methodology [Springmeyer, 92].

The process-oriented methodology is a three tier approach not intended for sequential execution. Activities within and across each tier were iterative and were conducted in parallel with the first tier, **General Process**. The General Process includes activities for observing and describing the scientific data analysis process. Interviews were conducted to become familiar with analysis methods and environments of scientists from different sub-disciplines and to identify candidates for observations sessions. The observation sessions were conducted such that the designer assumed the role of an “apprentice sitting in on expert’s work session in order to listen, question, and learn” [Springmeyer, Blattner, Max, 92]. The scientists were asked to proceed with their current analysis problems while talking and “thinking aloud”. The observation sessions were audiotaped to provide a permanent record of the session. Notes were taken to capture the scientists’ actions and

examples of work were collected. Each observation session was transcribed and summarized noting the main focus of the analysis and the observable actions. The summaries and transcripts were used to identify individual actions and list them in order of occurrence. As actions within and across observation sessions were compared and contrasted, categories of actions were developed. Fifteen fundamental elements (activities common to all observation sessions) were formed and grouped into four main categories. The purpose of decomposing actions into categories was to form a comprehensive overview of important, fundamental elements not a comprehensive category of all actions. Any scientific data analysis can then be described as a collection of these few basic elements. The decomposition also helps software designers become more familiar with a process they are unfamiliar with.

The second tier, **Implications for Design**, took into consideration how current technologies could affect the design of new tools. An action-versus-purpose framework was devised to examine how the scientists interacted with their data analysis environments. This framework demonstrated how actions can be accomplished in several different ways and how elements were currently supported (with or without the use of a computers). Determining how elements were supported uncovered new possibilities for applying technology to support both traditional and often neglected elements of scientific data analysis. The fundamental elements, decomposition, and the framework provided a frame of reference for addressing issues such as the effectiveness of current tools, how to improve the tools, and how new technology could be incorporated to assist scientists in analyzing data. Finally, design recommendations were suggested to guide the

development of a prototype tool and for the use in future development of tools for scientific data analysis.

In the final tier, **Specific Prototype Tool**, Springmeyer designed and implemented a prototype tool for data culling, an important element of scientific data analysis shown by her study to be inadequately supported. She observed scientists culling data with this prototype tool, and then extended the decomposition of scientific data analysis to include elements of data culling.

Springmeyer's approach is very similar to those of Contextual Inquiry [Duncan & Beabes, 95] and Contextual Design [Holtzblatt & Beyer, 1995]. Contextual Inquiry is a set of data gathering and analysis techniques designed to provide designers with grounded and detailed knowledge of the users' work. The key concepts behind Contextual Inquiry are:

- **Context:** understanding the users' needs in their work environment.
- **Partnership:** making users co-investigators in the design process by asking the users not the designers to interpret the users' actions.
- **Focus:** designer's assumptions and beliefs are aligned with the users' through contextual inquiries.

Contextual Inquiry data gathering techniques include work-based interviews similar to Springmeyer's designer-as-apprentice interviews, post-observation inquiries, artifact walkthroughs, future scenarios, and prototype/test drives (see Glossary for a definition of

terms). These techniques are designed to meet the needs of the work environment with minimal disturbance. For instance in work environments where the tasks are time critical (*e.g.*, emergency rooms), the users cannot stop to answer questions or think/talk aloud as in work-based interviews. Post-observation inquiries allow the designer to observe the time critical task and take notes based on observations. After the task is completed and the criticality of the situation passes the designer can review his notes and questions with the users to gain more information about the users' intent.

The data gathered from these techniques can be structured using a bottom up process, Affinity Diagrams, to identify key design ideas, new questions for users, technology to explore, and the next interview participants. Affinity diagrams help designers to organize the data and to capture all the details. Contextual Design's work models can then be used to capture and coherently represent the details of how users' work. Work models are "simple drawing techniques that describe the most basic aspects of work relevant to design: the culture and physical environment it takes place in, the pattern of human interaction and communication, and the basic tasks and steps involved in getting the work done" [Holtzblatt & Beyer, 95]. There are four work models to represent each of these aspects: context, physical, flow, and sequence models. Design issues can be raised by walking through these models with the design team and users and asking questions such as "can we combine and eliminate steps?" The design issues are then addressed in the design of the system that will eventually redesign the work practices of the users.

Springmeyer's process-oriented methodology, Contextual Inquiry, and Contextual Design answer the question of how designers learn about users' work. They also enhance traditional analysis methodologies when used together in that they:

1. gather immediate data from the users about their actions and intents not relying on just users' recall. These methodologies recommend several techniques for gathering data to alleviate the bias from using just one technique.
2. focus on understanding the process and not on the development of any one tool. Tools that better fit the user's needs can then be conceived of from the results.
3. involve the users throughout the process allowing for more timely input of information than just at the data gathering and model verification stages.

Chapter 3 describes how these methodologies were applied to the specific case of studying the process of setting up and solving nonlinear finite element problems.

2.2 Current Mesh Tools

This section provides a general description of a mesh generator, an unstructured rezoner, and a structured rezoner. The NGP mesh generator, like many mesh generators, provides users with the capability to check the quality of a mesh and fix it. These capabilities help users build the initial mesh which may be good in the sense that some user-defined measures of quality are met. However, a good mesh must also reflect the structural behavior of the problem. Interactive-adaptive³ methods such as unstructured and

³ [Gattas & Abel, 83] use this term to refer to methods that permit analysis parameters and algorithms to be selected and/or changed by the user during the analysis itself.

structured rezoners discussed in the following sections actively promote the users' understanding of the problems' behavior and provide capabilities to monitor the results and intervene and control the analysis in real computing time. These tools are described here in order for the reader to become familiar with the state-of-the-art mesh generating and rezoning tools. These tools or similar tools were used by the scientists participating in this study. Familiarity with these tools will assist the reader's understanding of the results of the study described in Chapter 4 and explain the need for additional tools to support this process.

2.2.1 Mesh Generator

The National Science Foundation Engineering Research Center at Mississippi State University developed an interactive/automated grid generation system, NGP, which provides a uniform environment for the construction of computational grids using state-of-the-art technology [MSU/ERC, n.d]. Users can construct a geometry representing the problem domain using Computer Aided Design (CAD) and surface generation capabilities built into NGP or some other system (*e.g.*, a CAD package such as ProE) and use it as input into NGP. NGP can mesh both two and three-dimensional geometry. The two- and three-dimensional structured meshes are made up of quadrilaterals and hexahedra, respectively. Unstructured surface grids of triangles and unstructured volume grids of tetrahedrons can be generated. NGP also provides tools for visualizing the quality of the mesh so that it can be fixed prior to running the solver. The user is provided with a

set of quality measures for structured surfaces and volumes, and unstructured surfaces and volumes as listed in Table 2-1.

<i>Structured Surface/Volume</i>	<i>Unstructured Surface/Volume⁴</i>
aspect ratio	aspect ratio
skewness	distortion
stretching	smoothness
	adjacent areas/volumes
	connectivity points
	solid angle
	angle minimum

Table 2-1. NGP mesh quality measures.

The user must specify a tolerance (*e.g.*, $\text{measure} = 0.5$ or $0.1 < \text{measure} < 0.6$) for each quality measure for the appropriate mesh type. All elements whose measures are less than or equal to a tolerance are displayed in one color while those elements not meeting this tolerance are transparent. The user can repeatedly refine these measures to find the areas requiring local refinement. Once these areas of the mesh are identified, the user must use the CAD and surface capabilities to modify the geometry and then regenerate the mesh.

NGP's grid quality visualization capability helps users develop better initial meshes. However, this capability is still limited. Measures that describe the quality of a mesh vary from problem to problem. With NGP the user cannot select or define the measures appropriate to his/her problem. He/she must specify a tolerance for all two- or three-dimensional quality measures depending on the problem's dimensions. This limits the users' ability to select the appropriate measures for their problems. To refine areas of the mesh the user must use tools specifically designed for building a mesh. Other mesh generators such as TrueGrid [XYZ Scientific Application Inc., 93] provide relaxation

⁴Solid angle and angle minimum are quality measures for surface grids only.

methods for improving areas of the mesh after it is built. Although the user might start his/her analysis with a better mesh, the mesh may change during the course of the simulation requiring additional modifications.

2.2.2 Unstructured Two-Dimensional Rezoner

NIKE2D is an implicit finite element solver with a built in analysis and rezoning capability [Engelmann, 91]. NIKE2D is an implicit Lagrangian code well suited for problems with moderate element deformation. At user defined intervals, graphical display of the results can be used to study the current state including the capability to display:

- color fringes,
- contour lines,
- vector plots,
- principal stress lines,
- deformed meshes and material outlines,
- profile plots,
- reaction forces,
- interface pressures along slidelines.

The display capabilities allow the user to view the results during execution. With this capability the user doesn't have to wait until NIKE2D has completed its execution before examining the results.

The rezoning capability extends the domain of applications for NIKE2D to problems with large deformation. Either interactively or automatically via a command file the solver can be stopped to rezone the mesh. The user identifies areas of the mesh for refinement by

visually inspecting the mesh or by analyzing the state data with the capabilities described above. The rezoner provides the following capabilities for refining the mesh:

- reset run-time parameters;
- select the area to be smoothed by material type or by arranging the area within the display window;
- select the smoothing method;
- smooth the mesh;
- restore original mesh when changes do not improve the mesh;
- move a selected node;
- change the spacing of nodes on a boundary;
- dekind slidelines, boundaries, and sides.

The rezoner provides three methods for smoothing interior nodes: the equipotential method, isoparametric interpolation, and a combination of the two obtained by a linear blending as described in [Engelmann, 91].

The rezoning capability is used infrequently and only by the most sophisticated users. A NIKED2D developer speculates that is due mainly to the users' focus on solving the problem and not on exploring the capabilities of NIKE2D. It may also reflect on the usability of the tool. The rezoner has a command line interface where the commands ranging from 1 to 3 characters in length are difficult to remember. The commands are also non-intuitive and little on-line help is provided. Table 2-2 illustrates a short sequence of rezoning commands.

<i>Command</i>	<i>Description</i>
> control C	Interrupt solver.
> sw 5	Start rezoner.
> g	Display mesh.
> m 1	Select mesh defined as material 1 to smooth.
> s	Smooth mesh with material 1 properties.
> s	
> s	
> s	
> s	
> s	
> m 1 b	Display material one boundaries.
> vs 33 20 0.2	Vary the spacing between nodes 33 and 20.
> s	Smooth mesh with material 1 properties.
> s	
> s	
> s	
> s	
> s	
> blend 1	Select smoothing option to combined blending.
> s	Smooth mesh with material 1 properties.
> s	
> s	
> s	
> s	
> s	
> f	Terminate interactive phase.

Table 2-2. A sample command sequence for NIKE2D rezoning.

Identifying areas for refinement in a two-dimensional mesh relies on the user's ability to visually inspect the mesh and state data. Visual inspection of the mesh works for two-dimensional meshes because the user can see the entire mesh. It will not work for three-dimensional problems because the user can't see the entire mesh without additional tools. Additional tools are also required for selection of a three-dimensional region. Centering a region of the mesh in the display window (as is done in NIKE2D rezoner) or using the mouse to graphically select an area works for two-dimensions but not three. Two-

dimensional rezoners, especially the NIKE2D rezoner, cannot be easily extended to support three-dimensions for these reasons.

2.2.3 Structured Rezoner

[Lu & Eiseman, 91] developed a grid quality manipulation system for two- and three-dimensional structured meshes used in computational fluid dynamics problems. This system allows users to visualize mesh quality measures and interactively manipulate the grid. It provides the capability to visualize quality measures directly derivable from the grid elements. This constitutes the measures of coordinate curve lengths, intersection angles, cell area, and conformality. These measures are reliable only when the solution of the physical problem is mildly varying with respect to the arc lengths in the physical region. Quality measures based on solution attributes are necessary when rapid solution variations are present. In this system a monitor surface [Eiseman, 85] represents a basic means for measuring curvature. Curvature gives information about the severe bends or direction changes in the surface. The monitor surface is the basic link to the data generated by the solver. The user can select any of these quality measures to be displayed as a color graphic image. The cells are colored based on values of the measure and then the grid is laid on top of the solid colored region. The color variations indicate where the measured quantity or quantities may point to problems in the grid.

A control net is placed on top of the grid. The control net is a sparse array of points. The points are placed using the control point form of algebraic grid generation [Eiseman, 88]

in order to represent the original grid. The user views the grid and the control net as sheets. The sheets are the coordinate or natural net surfaces that are given by holding fixed one of the three curvilinear coordinates. By shifting through the collection of sheets the user gains an understanding of the problem and can locate "bad" areas of the mesh. The mesh can be altered by moving the control points. The control points can be moved within the net sheet surface (movement on the plane) or along the transverse links to the control point just above or below the net sheet surface (movement off the plane). As the control point is moved, both the grid and the quality measures are updated so that the consequences of the user's manipulations are continuously seen.

This approach works for structured meshes because a structured mesh can be thought of as an array of elements. An element's neighbors can be identified by incrementing the array indices. Shifting through the sheets is just looping through the indices of the array. The concept of shifting through sheets in an unstructured mesh breaks down because an unstructured mesh represents the connectivity of the mesh in a linked list fashion where each element points to its neighbors. Here there aren't any indices to loop through and therefore no sheets to represent the grid or the control net and points used for grid manipulation.

Chapter 3

Observations and Analysis

This chapter describes how elements of process-oriented methodology, Contextual Inquiry, and Contextual Design were applied to the study of solving nonlinear finite element problems.

3.1 Interviews and Observations

Two design meetings were conducted with two groups of four mechanical engineers. These meetings were free form, open-ended discussions on the shortcomings of the current tools used in analyzing metal forming models and possible improvements for tools. Metal forming problems start with a very simple grid representing a sheet of metal to be formed as shown in Figure 1-1. This sheet is then molded into the desired shape. The areas of interest move as the sheet is molded, making it difficult to represent these areas in the initial mesh. At the conclusion of these meetings it was very obvious that new tools were needed to support this type of work and the users had several design ideas. Do any other areas of finite element analysis have similar requirements? And, what tools could be developed to support this work?

In order to answer these questions a better understanding of the work and information from a more diverse set of users was needed. Three pre-interviews were conducted with

scientists with interests in two- and three-dimensional high impact and implosion problems. These pre-interviews consisted of a fixed set of questions asked of each scientist. Appendix A lists the questions asked during these interviews. The results of these interviews confirmed that the issues discussed above were issues in other disciplines and worth pursuing. Finally, six scientists at Lawrence Livermore National Laboratory were interviewed using the work-based interviews and artifact walkthroughs. One of the six participants participated in the design meetings. One of the pre-interviewee participants was also present at the work-based interviews. The participants consisted of tool developers as well as end users. Table 3-1 summarizes the participants.

<i>Field</i>	<i>Type of Interview</i>	<i>Topic of Interview</i>
Mechanical Engineer (end user & tool developer)	Work-based Interview	2-d metal forming problem using a 2-d rezoner.
Mechanical Engineer (end user)	Artifact Walkthrough Design Meeting	3-d thermal and structural problems.
Physicist in Physics Group (end user)	Interview	2-d and 3-d implosion problem.
Mathematician in Physics Group (end user)	Interview	3-d structural problems.
Mathematician in Physics Group (end user)	Artifact Walkthrough	3-d structural and fluids problems.
Computer Scientist supporting Physics Group (tool developer)	Work-based Interview	2-d implosion problem with 2-d rezoner.
Mechanical Engineer (end user & tool developer)	Artifact Walkthrough Interview	2.5-d metal forming problem.
Computer Scientist supporting Physics Group (tool developer)	Work-based Interview	2-d structural problem with 2-d rezoner.

Table 3-1. Summary of Interviews.

The process of setting up a mesh, running the solver, and analyzing the results can take anywhere from several hours to several weeks to complete depending on the complexity of the problem. Since it was impossible to observe the entire process, work-based interviews were focused on one or two iterations of the process: running the solver with an already developed mesh, reviewing the results, and fixing the mesh. Focusing on only this portion of the process during the work-based interviews captured the iterative nature of this work. Work-based interviews were used when the problems were small and the work was supported by two-dimensional rezoners.

Artifact walkthroughs were used to acquire information about the process not observed during the work-based interviews and for three-dimensional problems not supported by a rezoner. Artifact walkthroughs were also necessary when scientists were not solving finite element problems at the time of this study, but had recently completed an analysis of a finite element problem. For an artifact walkthrough, the participants are asked to collect all artifacts (*e.g.*, notes, reports, forms, databases, letters) pertaining to the process under study. The participants then reconstruct the process talking aloud while the observer takes notes and asks questions. The artifacts help the participants recall information about the process. In this case, the scientists gathered their notes, documents, meshes, and analysis results from a particular project involving the analysis of a finite element problem. Often the scientists used software tools to graphically display and interrogate the mesh and the analysis results.

The same fixed set of questions used in the pre-interviews were also used in the work-based interviews and artifact walkthroughs. These questions provided a means for opening up the discussions with the scientists and giving them a time to relax as well as to obtain some preliminary data about their work. Notes were taken during interviews, work-based interviews, and artifact walkthroughs capturing as much of the dialog as possible as well as the scientists' activities and their interactions with tools such as print outs, computer applications, and manuals. Samples of scientists' work were collected, including manuals, and scratch paper on which quick calculations or examples were illustrated. The notes from these sessions were typed within a day of each session; additional details recalled about the process by the interviewer were added to the transcripts at that time.

Audio taping or video taping would have been preferable means of capturing data during these sessions although slightly more intrusive. However, special permissions and procedures are required for the use of audio and video equipment in the areas of Lawrence Livermore National Laboratory where the participants were located. Therefore, some of the details were lost in order to conduct the interviews in an easy and timely fashion. However, enough details about the process of solving finite element problems were gathered in order to improve the design of future tools.

3.2 Analyzing the Data

Techniques from both Springmeyer's process-oriented methodology and Contextual Design were used to organize and understand the data collected from the sessions described above. Each session was summarized in order to become more familiar with the observation session. The session from the superplastic metal forming problem in Figure 1-1 is summarized in Figure 3-1.

Summary

(A short description of the format of the interview, the problem being studied, and the software applications being used. The summary was followed by questions which focused on the area of interest for this study.)

There were two parts to this interview:

1. The first part was a reconstruction of a 2 1/2 D problem where a piece of sheet metal was pressed into a pan shape.
2. During the second part we talked about different types of 3D problems for rezoning. It became obvious that the only time you are interested in rezoning/remeshing is where there is a lot of deformation to the mesh such as in metal forming or forging.

How was the original mesh created?

3D meshes can be constructed using mesh generators such as INGRID, TrueGrid and NGP. Some mesh generators such as TrueGrid and NGP accept input from CAD packages. In these cases drafters can build the parts and feed them into a mesh generator. Sometimes the parts are defined as a blueprint such as a submarine hull or car. Parts can also be defined with measurements.

How did the user know when the mesh was not good enough (i.e., needed to be modified)?

By looking at the areas of interest you can tell if the mesh is good enough. Looking means zooming, rotating, and displaying parts together or separately. After the first problem ran to completion the contour intervals for thickness were close in value to the experimental data. This said that NIKE could be used to analyze this problem. The next step was to make the mesh better in areas of interest, namely the corners. You could tell that the mesh wasn't very good in the corners because the mesh didn't fit into the corners. Taurus was used to analyze the results. For 3D problems you may need to look at how skewed elements are, aspect ratio of nodes and the shearing of elements to determine if the mesh is good. TrueGrid has a list of quality measures. The ideal is to have square elements.

When did the user know if the mesh was not good enough?

Some solvers such as NIKE3D and DYNA3D do not have any way to visualize the results concurrently with execution. So results must be viewed after the solvers have completed using post-processing tools such as Taurus and GRIZ.

How did the user reconstruct the mesh in order to make it better?

You start by building the original mesh with enough detail in areas that you think are important then run the problem. Using a graphical post-processor view the resulting mesh to identify critical areas where the mesh can be improved. Modify the original input deck to reflect the changes. The input deck is edited with an editor such as vi and viewed by a pre-processor such as INGRID. While editing add parameters to make the changes to the mesh easy and centrally located in the input deck. Sometimes it is necessary to make a sketch of the drawing with the labels so in future modifications it is easy to remember which parameter should be changed. Modifications are really just guesses. Rerun the problem and continue until the results are acceptable. Ideally you would like a remesher to use some criteria entered by the user to determine when the solver should stop and rezone the mesh. The criteria for the first mesh would be curvature of the elements in excess of 15%.

Figure 3-1. Summary of an observation session.

The first step in analyzing the data was to identify, categorize, and decompose the process using the data and summaries. To do this a list of actions and their order of occurrence

was developed for each observation session. The list of actions were compared and contrasted, grouping similar actions by user's intent. Figure 3-2 displays the decomposition.

* Script Changes pertain only to solving two-dimensional finite element problems. Tools for three-dimensional finite element problems did not support scripting of any sort.

Figure 3-2. Decomposing the process into categories of related actions.

The decomposition contains three main branches of actions (investigation, manipulation, and comparison) and six subcategories (examine mesh, select mesh area, modify mesh, compare results, view results, script changes). **Investigation** refers to the scientists' exploration of the mesh to obtain information about the "goodness" of the mesh. "Goodness" of a mesh, more frequently referred to as mesh quality, is a subjective measure depending on the problem and directly relates to the accuracy of the solution. Curvature was the measurement of quality in the superplastic forming analysis example in Section 1.1. Quality measures used by scientists in this study included aspect ratio, the one-to-one matching of nodes between parts, skewness of elements, shearing of elements,

and curvature. Scientists determined the quality of the mesh through examination of state and derived data (**View Results**) and visual inspection of the mesh (**Examine Mesh**).

Manipulation categorizes the actions a scientist takes to change the mesh in hopes of developing a more accurate mesh. In the case of two-dimensional problems, software was available in either the form of a stand alone program or integrated into the solver to support scientists' manipulation of the mesh. This software, called a rezoner, permitted scientists to move nodes, smooth boundaries, smooth mesh regions, add nodes, delete nodes, and undo any or all changes (**Modify Mesh**). In the case of three-dimensional problems, scientists modified the original mesh using a mesh generator. Actions were limited to moving, adding, and deleting nodes. Actions to modify a mesh all started with the selection of an area of the mesh to modify (**Select Mesh Area**). Individual nodes, a region (*i.e.*, a group of elements not bounded by a particular material), a material (*i.e.*, all elements with a common composition or material type), or a boundary could be selected for modifications. Actions to modify the mesh could also be scripted so that these same actions would be executed in future runs of the solver (**Script Changes**). Script changes were only possible with the support of two-dimensional rezoners.

Comparison refers to the actions a scientist takes to determine if the newly modified mesh is better than the original mesh. Many of these actions are similar to those in the Investigation category, namely the visual inspection of the mesh (**Examine Mesh**). It also includes comparing the results of one mesh with the another (**Compare Results**) by quickly flipping between images or comparing tabular state or derived data.

Table 3-2 lists actions associated with the six subcategories.

<i>Subcategory</i>	<i>Actions</i>
Examine Mesh	Zoom Pan Rotate Display by material type
Select Mesh Area	Pick a node Specify a material type Select a region by positioning display window around the region Specify a boundary
Modify Mesh	Move nodes Smooth boundaries Smooth areas Add nodes Delete nodes Undo changes
Compare Results	Compare derived results Flip between images
View Results	Fringe plots Contour plots
Script Changes	List commands

Table 3-2. Actions associated with each subcategory.

The decomposition of fundamental elements⁵ encompassed both two- and three-dimensional finite element analysis, indicating that the basic activities for the two processes were the same. While developing this decomposition it became obvious that how scientists accomplished these actions were different in two-dimensions than in three. This difference suggested that a comparison of the tools and the processes for solving two- and three-dimensional finite element problems was needed. Physical and sequence work models from Conceptual Design were used to assist with this comparison.

The physical model as shown in Figure 3-3 identifies the software tools that scientists use. Many of the software tools produce files that are used by other tools. The main

⁵ Springmeyer refers to an action important and common to all sessions as a fundamental element.

difference between the sets of tools is that rezoners for two-dimensional tools exist while three-dimensional rezoning tools do not exist or were not available to these users. Other tools used include calculators to check intermediate results, paper to track notes or annotate intent, and users manuals. Users manuals were frequently consulted because commands were short and cryptic, making them hard to remember.

Figure 3-3. Physical model for solving finite element problems.

A sequence model represents a user's actions and the order of those actions by which the user's intents are achieved. Actions are connected by arrows to represent order, loops, and branches in the process. The user's intents are marked next to an action or group of actions; problems are marked with $\mathcal{N}$. A sequence model was developed for each work-based interview and artifact walkthrough. All of the sequence models for solving three-dimensional finite element problems were basically the same and are represented by one model depicted in Figure 3-4. The sequence models for solving two-dimensional finite

element problems varied quite a bit because the tools provided different ways to accomplish the same actions. These models were consolidated such that the key actions and intents from all models are represented (see Figure 3-5).

Figure 3-4. Sequence model for three-dimensional finite element problems.

element problems, rezoners exist allowing scientists to run their problems until the solver quits, evaluate each state, find the *earliest* state where the mesh exhibits problems, modify the mesh at that state, then start solving the problem from this state with the new mesh. In this process the scientist does not have to guess at where to modify the original mesh; he/she can modify the mesh at the state where the problem seems to arise. Scientists can also modify the mesh several times during the course of one run. The modifications can be saved so that the solution can be reproduced.

Chapter 4

An Interactive 3D Rezoner

The process study in Chapter 3 indicated that three-dimensional unstructured mesh redistribution is inadequately supported. The goal of the design presented in this chapter is to illustrate how the knowledge gained in the process study can be incorporated into the design of software systems. Although the entire finite element problem solving process could be better supported, the effort required to produce a system which addresses all aspects of three-dimensional rezoning is beyond the resources available to one researcher. For this reason, two fundamental elements inadequately supported in three-dimensions were chosen for prototype development: mesh investigation and manipulation. Specifically the prototype tool supports the examination of a three-dimensional mesh in order to determine the quality of the mesh and the selection of "poor" quality areas for manipulation. Section 4.2 describes in detail the prototype tool.

4.1 Design

Scientists need a system that provides the capability to manipulate three-dimensional, unstructured meshes at any state during the analysis of a nonlinear finite element problem. According to the decomposition of this analysis process, shown in Figure 3-2, this system must support three main branches of actions: Investigation, Comparison, and Manipulation. Investigation refers to the scientists' exploration of the mesh to obtain information about the "goodness" of the mesh. Scientists currently determine the quality

of the mesh through examination of state and derived data and visual inspection of the mesh. Many post-processors currently provide scientists with these capabilities. Since these capabilities already exist it is not necessary to reinvent the wheel when designing a rezoning capability. This integration of the post-processor functionality with the solver is natural and has been attempted by others [Gattass & Abel, 83; Engelmann, 91; Kerlick & Kirby, 93]. The design then consists of four modules: a controller, the solver, the post-processor, and the rezoner. The diagram in Figure 4-1 depicts the preliminary design in a data flow diagram format.

Figure 4-1. Three-dimensional Rezoner Design.

4.1.1 Controller

The controller is a software module controlling the execution of the solver. The controller takes commands interactively from the user or from a command script. The commands allow the user to interrupt the execution of the solver for visualizing analysis results, examining mesh qualities, and/or manipulating the mesh. The command script is initially constructed by the scientist. As the scientist successfully manipulates the mesh, the commands can be added to a script in order to regenerate the results of the analysis at a later date. The script can also be used with a slightly different mesh in order to support mesh resolution studies.

4.1.2 Solver

Solvers generate plot files at each state. A plot file contains node locations, connectivity data, and state data. The format of these plot files are solver specific. In this design, the solver can be any three-dimensional finite element code that generates plot files in "MDG plot file" format. The post-processor selected for use in this prototype accepts only plot files in this format.

4.1.3 Post-processor

The post-processor provides capabilities with which the users are already familiar: viewing results and examining the mesh. GRIZ [Dovey & Spelce, 93; Dovey & Loomis, 94] was selected as the post-processor for this prototype. GRIZ is a general purpose post-

processor designed to work with a variety of finite element solvers for unstructured grids; namely, DYNA3D, NIKE3D, and TOPAZ3D [Shapiro, 85]. It provides support for modern three-dimensional visualization techniques for the exploration and investigation of analysis results. These techniques include isosurface displays, cutting planes, vector grids, particle traces, fringe plots of result values on the surface of a mesh, contour plots, and volume rendering techniques. Capabilities exist for rotating, translating, and scaling the view of the mesh using the mouse; computing result minimums and maximums over all the states; setting cutoff values for the results display; and turning on and off the display of individual materials in the mesh. GRIZ also incorporates the capability to animate data over time and to store animation frames to video disk. This post-processor is designed to utilize the capabilities of Silicon Graphics workstations that provide hardware support for three-dimensional graphics. GRIZ uses OpenGL (graphics library) for rendering and the Motif widget toolkit for its user interface. A menu is provided for accessing a subset of the most frequently used GRIZ commands. A command line interface is available for typing in the full set of GRIZ commands.

4.1.4 Rezoner

From the investigation of the mesh, the scientist identifies regions in the mesh that need to be refined. The location of these regions and commands for modifying the mesh are passed to the rezoner along with the node locations, connectivity data, and results (state data). The commands for modifying a mesh should at a minimum include those actions identified in the process study and listed in Table 3-2. Those actions include: moving

nodes, smoothing boundaries, smoothing areas, adding nodes, deleting nodes, and undoing any and all modifications. A variety of algorithms exist for smoothing boundaries and areas [Thompson, Warsi, Mastin, 85; Thomas & Middlecoff, 80]. The rezoner must also map the results from the old mesh to the new mesh [Dukowicz & Padial, 91]. The new node locations and state data will be fed back to the post-processor so that the user can compare the resulting mesh with the old mesh. If the user accepts the changes, these changes will be fed to the solver and execution can begin again from this modified state.

4.2 Prototype Tool

As the process study progressed, it became clear that the tools used in actions categorized as *Investigation* in Chapter 3 are insufficient for three-dimensional problems. The Investigation category is composed of two subcategories: *view results* and *examine mesh*. As discussed earlier in this chapter, viewing of results seem to be supported sufficiently through post-processors. The process study categorized the actions of rotating, translating, and scaling the view of the mesh as the subcategory *examine mesh*. These actions are also supported by the post-processor. Two-dimensional meshes and surface meshes are easily examined with the current tools. However, examination of the interiors of three-dimensional meshes require more advanced interaction techniques. These actions are also performed when users *select mesh areas* for manipulation; in a two-dimensional rezoner the area to be smoothed must be centered within the display window or be of one material type. Obviously a three-dimensional area cannot be adequately selected using a

two-dimensional window. Another technique is needed for selection of a three-dimensional areas.

The post-processor GRIZ has been extended to better support three-dimensional mesh examination and selection of mesh areas through the visualization of quality measures, the concept of peeling, and mesh quality queries. Quality measures are measures of the "goodness" of a mesh. They can either pertain to the shape of the mesh elements or to finite element results (*i.e.*, state data). Peeling is a concept of pulling off layers of the meshes in order to view the interior of the mesh.

The process study findings indicated that quality measures are problem dependent and can be either derived directly from the mesh (*e.g.*, aspect ratio) or from the state data (*e.g.*, pressure). GRIZ currently supports the visualization of state data with the results command. The user selects the result variable to be displayed and the results are color-mapped with regions of the highest result value drawn in the color at the top of the colormap, regions of the lowest result value drawn in the color at the bottom of the color map, and so on. Color-mapping of results is a very common and successful technique for visualizing results.

The prototype tool expands the GRIZ results command to display quality measures directly derivable from the mesh [MSU/ERC, n.d.; Stewart & Abolhassani, 93]. Two quality measures were implemented in the prototype tool: aspect ratio and skewness. Appendix B lists the code for generating these quantities. Figure 4-2 displays skewness in a three-dimensional metal forming problem at the last state. The color of each element reflects the value of skewness. With this capability the user can move between states and

examine the changes in quality over time. However, the user still cannot easily visualize the interior of the mesh.

Figure 4-2. Skewness of Three-Dimensional Model.

The concept of peeling was implemented to assist the user in examining the quality of the interior of a mesh. Peeling is built on top of GRIZ's rough cutting plane display. The

“rough cut” option deletes all elements that intersect the cutting plane or are on the side of the cutting plane in which the plane normal vector points. This is an effective technique for “seeing” inside a three-dimensional model. The concept of peeling is to start from one side of the model and peel layers (or planes in this case) off the mesh until the other side of the mesh is reached. The user defines the location of the initial cutting plane, the location of the ending cutting plane on the other side of the model, and the number of peels. Table 4-2 describes the peeling commands.

<i>Command</i>	<i>Description</i>
peel x <i>n</i>	Using the models limit box, start from the minimum x value defining <i>n</i> peels ending with the maximum x value. The cut plane is perpendicular to the x axis.
peel y <i>n</i>	Using the models limit box, start from the minimum y value defining <i>n</i> peels ending with the maximum y value. The cut plane is perpendicular to the y axis.
peel z <i>n</i>	Using the models limit box, start from the minimum z value defining <i>n</i> peels ending with the maximum z value. The cut plane is perpendicular to the z axis.
peel <i>x_f y_f z_f x_t y_t z_t n_x n_y n_z n</i>	The initial cut plane is defined by the point <i>x_f y_f z_f</i> on the plane and the vector normal (<i>n_x n_y n_z</i>) to the plane. The ending cut plane is defined by the point <i>x_t y_t z_t</i> on the plane and the same vector normal.
on/off rough	Turns on (or off) the rough cutting plane display.
peel n <i>n</i>	Peel <i>n</i> more layers.
peel p <i>n</i>	Replace the last <i>n</i> layers (reverse peeling).
peel off	Delete the cutting planes defined for peeling and return display to original condition.

Table 4-2. Peeling Commands.

Figure 4-3 depicts a series of peels in a three-dimensional model where a ball is pressed through a series of plates. The colored elements display element skewness. The skewness of the elements on the surface look good. However with this technique interior regions

are identified where the skewness of elements are not as good as those at the surface. In this particular example skewness is still quite good and does not require any mesh manipulations since there is little deformation. However, in other problems with larger deformation the user could then take action to correct the mesh once these "bad" regions were identified.

solid shell-solid-shell=col
t = 5.49931e-04

solid shell-solid-shell=col
t = 5.649431e-04

solid shell-solid-shell=col
t = 5.849431e-04

Figure 4-3. Peeling with Skewness.

A method is required for selecting regions of the mesh to be fixed. The GRIZ menu was extended to provide a rezoning menu as displayed in Figure 4-4. The user defines the tolerance of the mesh qualities, then he/she defines the query by defining a boolean expression using mesh qualities and boolean operators. The query in Figure 4-4 reads: find all elements with aspect ratio greater than 0.5 and skewness greater than 0.75. All elements satisfying this query will be highlighted as shown in the graphics window of Figure 4-4. The user can use the GRIZ selecting and picking commands to add or delete specific elements from the selection. All selected elements can be passed to the rezoner for manipulation.

Figure 4-4. Quality Query Menu and Results.

Chapter 5

Rezoner Evaluation

Prototyping allows potential users to test and/or evaluate new systems and user interfaces early in the product development life cycle. Results of the users' tests of a prototype tool can be fed back into the development process to ensure that a tool meets the users' expectations and needs. Detailed observations of the users' use of the prototype tool can be gathered using the techniques described in Chapter 3. This chapter discusses how the observation techniques were used to evaluate both the design and the prototype tool, and presents the results of this evaluation.

5.1 Evaluation Process

Two evaluation sessions were conducted. The first evaluation session was conducted to evaluate the design for a three-dimensional rezoner as described in Chapter 4. The second evaluation session was conducted to evaluate the prototype tool also described in Chapter 4. For the evaluation of the design, one scientist and one software developer were selected to participate in the evaluation session. The scientist participated in the earlier observation sessions and was selected because of his experience with two-dimensional rezoners. The software developer is the developer of GRIZ and did not participate in earlier observation sessions. He was selected because of his knowledge of GRIZ and its use. The same scientist participated in the prototype tool evaluation. This scientist was selected for the prototype tool evaluation because of his inexperience with GRIZ. The scientist had used GRIZ on several occasions but was not considered an expert. The level

of expertise with GRIZ was important in order to evaluate the intuitiveness of the enhancements made to GRIZ for the prototype tool.

The evaluations were conducted using Contextual Inquiry's Prototype/Test Drive method [Duncan & Beabes, 95]. The Prototype/Test Drive method consist of four steps:

1. The observer asks the participant to talk about his/her work, problems, scenarios, or typical occurrences.
2. The observer shows the participant the prototype tool.
3. The participant is asked to talk aloud about what he/she would do, what he/she expects to see, if his/her expectations were met, and if not what should have happened. The model in Figure 5-1 describes this interaction between the participant and observer. The observer notes the participants actions as well as his/her reactions.
4. The observer and participant change the prototype tool "on the fly".

Figure 5-1. Prototype/Test Drive Cycle [Duncan & Beabes, 95]

The evaluations were conducted in the participants own environments, their offices. To begin with the participants were asked to talk about three-dimensional rezoning, problems with the current process, and their most recent experiences with three-dimensional

problems requiring mesh modifications. This discussion validated the initial findings (see Chapter 3) and brought forward issues involved with three-dimensional rezoning. Ideas and problems identified in the earlier observation sessions were shared with the participants. Following this discussion the overall design for a three-dimensional rezoner depicted in Figure 4-1 was presented to the participants and the participants were asked for their reactions. Their reactions were noted and as questions were raised the observer and the participants modified the design.

For the prototype tool evaluation, the evaluation session started with a short description of the prototype tool. The scientist was also presented with a short description of each of the tool's commands (see Appendix C). After reading the short description, the scientist was asked to exercise the prototype tool on a problem of his choice. When problems or breakdowns were encountered the observer and scientist worked together to redesign the interface or tool. The white board was used frequently to sketch examples and proposed changes to the design.

5.2 Design Evaluation Results

Figure 5-2 depicts the modified design; with changes based on the input from the evaluation. The first area of confusion was with the controller. The function of the control remained the same but it was agreed that the controller should interact with the rezoner and the post-processor too. The controller should be the process that starts up and kills each of these applications. The picture led the participants to believe that the solver

would start up the rezoner or the post-processor. Also the picture led the participants to believe that to do rezoning the post-processor was always necessary. So the design was modified so that scripted commands were passed to the rezoner as well as the solver. These scripted commands would consist of an identification of a mesh area and the manipulation commands to be performed on that area. The post-processor would pass these same commands to the rezoner when the user wanted to interactively direct the control of the rezoning.

The second area of confusion was the functionality of the rezoner. Figure 4-2 depicted the rezoner as actually modifying the mesh and mapping the results from the old mesh to the new mesh. Remapping is an optional process and was added to the data flow diagram as a separate process.

Figure 5-2. Modified Three-dimensional Rezoner Design.

5.3 Prototype Tool Evaluation Results

For the prototype tool evaluation the scientist selected the simple ballplot example illustrated in Figure 4-3 to test the prototype tool. In this problem a ball is pressed through a series of plates. The plate elements in contact with the ball are slightly deformed as the ball is pressed through. Although the deformation is small in this particular problem, the problem is well understood. So it was a good exercise for this evaluation because the scientist could then concentrate on the use of the tool versus the analysis of a complicated three-dimensional problem.

The scientist started by first displaying skewness and aspect ratio. The scientist was able to display these results without hesitation. Treating quality measures as another result to be analyzed was natural and intuitive to the scientist. This was not surprising since the scientist was familiar with the results analysis capabilities of GRIZ and analyzing quality data was a natural extension. What was surprising was the scientist's use of time history plots. GRIZ provides the capability for a result such as stress to be plotted against time for a selected set of elements. Figure 5-3 displays a time history plot of aspect ratio versus time for four selected elements on the plate near the ball. The scientist used time history plots to validate the color display of aspect ratio. In the plot he expected the aspect ratio for the selected elements to decrease (get worse) over time. However the plot indicated that the aspect ratio increased (got better) after it decreased. After zooming in on that area of the mesh, selecting neighboring elements for time history plots, and animating through the states, it became obvious that as the ball pressed through the plates

the elements on the plates are squeezed making them thinner until the ball ruptures the plate; the elements thickened as the ball presses through the plate. This capability to view quality measures as results highlighted aspects of the problem not formerly identified by the scientist. The scientist found this portion of the prototype tool very effective and intuitive.

Figure 5-3. Time History Plot of Aspect Ratio.

The scientist used peeling to examine the interior of the ball mesh. When using the peeling command, he expected to be able to automatically progress through each of the peels similar to the **animate states** command. The **animate states** command slowly displays state one, then state two, and so on until the last state is displayed. A user can

interact with the display (*e.g.*, rotate the model or stop the animation when something interesting catches his/her eye) while GRIZ animates the states. Users can manually move forward or backwards through the states using the menu options **first state**, **last state**, **next state**, and **previous state**. The prototype tool provided commands for manually moving between peels but included no commands for automating the movement between peels and moving to the first and last peel. The scientist suggested that these commands be added and that a pulldown menu be provided for peeling similar to the animation pulldown menu.

Several problems were found with the quality queries. First, as originally designed, the user specified exact values for the quality measures specified in the query. An example of a query is "Find all elements with aspect ratio less than 0.2". The scientist preferred to define what he referred to as a "quality function" that could identify areas where the quality of the mesh changed by some relative amount. For example, find all elements where the aspect ratio is 30% worse than its initial value. The quality function should also include changes in state data as well as changes in quality measures derived directly from the mesh. Figure 5-4 is the menu as redesigned by the scientist and observer during the prototype tool evaluation.

Quality Function:		
Aspect Ratio less than 30.0% and X Stress greater than 2.5		↑ □ ↓
Results::	Operators:	Values:
Aspect Ratio	less than	#
Skewness	less than or equal	###.#
X Stress	equal	###.##%
Y Stress	greater than	
Z Stress	greater or equal	

Figure 5-4. Quality Menu Redesign.

Finally, elements that satisfy the conditions of the query are displayed by outlining their edges. This same technique is used for selecting elements for time history plots. This dual use of outlining elements confused the scientist. After discussing this with the scientist it was determined that elements that meet the query should be opaque and those that do not should be transparent.

Overall the scientist felt that this prototype tool was a “good first step towards rezoning three-dimensional (unstructured) meshes.” The prototype tool provided the capability to identify which state needed to be modified and where to refine the mesh. It also provided a mechanism for selecting areas of the mesh that were more intuitive than the tools currently used for two-dimensions. These tools combined with refinement algorithms alleviate the users’ need to return to the mesh generator to locally refine the mesh. The scientist also felt that these tools provided the user with the capability to “understand how mesh quality changes during the course of a simulation.” With this understanding the

users could begin to develop quality functions specific to their problems. The quality function could then be used as an error estimator to automatically stop the solver when certain conditions exist. The user could then use the standard mesh manipulation actions identified in Chapter 3 to improve the quality of the mesh.

Chapter 6

Discussion

In summary, this thesis was a systematic design of new tools scientists can use when solving three-dimensional, unstructured, nonlinear, finite element problems. The design process started with a study of how scientists currently solve both two- and three-dimensional, unstructured, nonlinear finite element problems. Through this study a description of the scientists work, the flow or structure of their work, a description of the problems or breakdowns in their work, and a description of the problems or breakdowns with their computer tools were constructed. A design of a three-dimensional rezoner and ideas for new tools to examine a mesh and select mesh regions emerged from this understanding of the users' work. A prototype tool was built providing users with the capability to color mesh elements according to a user selected quality measure, peel away layers of the mesh to examine three-dimensional meshes, and select mesh regions using quality queries. Techniques similar to those used for studying the process of solving finite element problems were used to evaluate the effectiveness of the design and the prototype tool.

The following sections discuss the advantages and disadvantages of the methodologies used for designing computer-based tools, and outline future work planned to support three-dimensional, unstructured, nonlinear finite element analysis.

6.1 Advantages and Disadvantages of the Approach

This thesis demonstrated how qualitative methods can be used to answer the fundamental question of how designers learn enough about the users' domain in order to develop software tools that support the users' work. These techniques support all phases of the software development life cycle. When used during the first phase of development (requirements specification) these techniques help designers collect information about the users' work, their actions and intents, the structure of the work, the tools (both computer and non-computer based) being used, what works well, and what does not. This information helps establish the requirements for a new system or modifications to the existing system. Designs and prototypes can be evaluated early in the design and implementation stages using these methods. The users act as co-designers defining what they think functions will or should do and ask for additional features as they walkthrough designs or work with the prototype tools. And of course, after the software is completed and delivered to the customer these methods can be used to measure the effectiveness of the product and to identify additional enhancements for future releases. In this study qualitative measures were used during the requirements specification stage and for the evaluation of the design and prototype tool.

Data gathering techniques from Springmeyer's process-oriented method and Contextual Inquiry provided ways to observe scientists unobtrusively in their work environments. These observation sessions were very useful because they provided additional requirements and ideas not identified in the design meetings. The design meetings were

focused on the development of a tool for solving 2 1/2-dimensional sheet metal forming problems. Observing scientists interested in the same problems as well as scientists interested in different problems broadened the scope of the problem to solving nonlinear finite element problems of which 2 1/2-dimensional sheet metal forming problems are a subset. They also provided a means for gathering specific details on how scientists work which led to valuable insight for the software designer. For example, the need for post-processing capabilities in the rezoner, comparing meshes by flipping between two images, and scripting mesh manipulations in order to reconstruct solutions⁶ are all ideas and requirements identified from the observation sessions. These kind of details about how scientists work are a crucial component in specifying requirements that lead to a better, more robust product that meets the needs of a wider range of scientists with similar problems.

Each data gathering technique discussed in this thesis is biased in some way. By using a combination of techniques, the biases are minimized. Traditional interviews and design meetings provide abstract data and very few details required for a complete design. They are constrained to the focus of the questions asked by the interviewer or designer and require the subjects to recall information. Work-based interviews alleviate the need for subjects to recall information. The subject adopts the role of a mentor providing accurate information about his/her actions and intent to an apprentice (designer) while working in his/her environment. This is a natural role for the subject and observer, requiring no

⁶See Chapters 3.0 and 4.0 for a complete list of findings from the observation sessions.

formal training. Other methods such as post-observation inquiries and artifact walkthroughs require subjects to recall information. However, these techniques are still superior to traditional interviews because the observer does not direct the users' conversations, thereby allowing more details about the process to be revealed. The use of notes or artifacts helps stimulate the subject's memory, increasing the accuracy of recalled information. Used together, these techniques provided a way to gather both general and detailed data from a representative sample of users.

Decomposition of actions, action-versus-purpose frameworks, affinity diagrams, Contextual Design work models, data flow diagrams, and information models are all techniques for understanding, organizing, and representing information about a process. This thesis used decomposition of actions, physical work models, and sequence work models. The decomposition of actions provided a useful way to categorize actions and help with the consolidation of sequence work models. The physical work model was fairly simple since this was a single-user process. It provided little additional information and could have easily been omitted from this study. The physical model may be of more use with more complex processes involving more people and/or artifacts. The sequence models provided an excellent way of diagramming the steps involved and comparing similar processes such as solving two- versus three-dimensional finite element problems. Organizing the data using these tools provides both graphical and textual representations easily shared with the users and design teams. As demonstrated in this thesis these tools complement the more traditional software analysis methodologies such as Object Oriented Analysis and Structure Analysis by providing the information needed to build

the conceptual models of the users' domain. The data flow diagram depicting the design of the new three-dimensional rezoner was built from the understanding gained through this process of studying users. It was also evaluated using these same methods of observation and user participation.

The same techniques for understanding the process were used to evaluate the design and the prototype tool. This interaction with the user provided the opportunity to improve the design and enhance the prototype tools during the design process. Users had the opportunity to participate in the design of new tools, thereby ensuring that the resulting tool would support their work.

In summary these qualitative methods are an important supplement to the software development life cycle. These methods form a suite of tools that designers can mix and match to gain an understanding of users' work and that can easily be scaled to fit the size of the project. In this thesis traditional methods (design meetings and interviews) were used in conjunction with participatory observations (work-based interviews and artifact walkthroughs) to gather data. A combination of techniques from process-oriented methods and Contextual Design were used to analyze and represent the data. The results of the analysis led to the design of a three-dimensional rezoner and the prototype of new tools to be used with redistribution algorithms. The design was depicted using data flow diagrams. Both the design and prototype tool were evaluated using these same user participatory techniques with the results of these evaluations leading to an enhanced design and prototype tool.

6.2 Future Work

At the conclusion of this thesis, the Methods Development Group at Lawrence Livermore National Laboratory has agreed to fund the development of the three-dimensional rezoner outlined in Chapters 4 and 5. The tools implemented in the prototype will be incorporated into the final product with modifications to reflect the suggestions gathered during the evaluation. Additional tools will be needed to support the other categories of actions identified in this thesis. These categories include Investigation, Manipulation, and Comparison, which were further subdivided into six sub-categories: View Results, Examine Mesh, Compare Results, Select Mesh Area, Modify Mesh, and Script Changes. The prototype tool suggested new tools to support actions of examining a mesh and selecting mesh areas. Suggestions for additional tools resulting from this analysis are listed below.

1. A new mesh can be compared with the original mesh by flipping quickly between these images or by overlaying one image on top of the other. Color can be used to highlight the differences in the two meshes.
2. The results of the quality query could be modified to use both color and opacity [Keller & Keller, 93]. Elements satisfying the query would be opaque and mapped into the color scale. Elements not satisfying the query would be transparent but outlined to help orient the user.
3. Cutaways and ghosting are effective techniques for providing a reference for locating features and for removing visual obstructions [Feiner & Seligmann, 92]. With these techniques, obscuring objects are rendered semi-transparent (ghosting) or partially cut away (cutaways) to reveal objects behind them. For three-dimensional rezoning the obscuring objects could be the outer layers of the mesh or "good" quality elements.

These techniques might be useful in examining or comparing three-dimensional meshes.

4. Scientists manipulate several states during the course of solving a finite element problem. Each of the scientists' commands to manipulate the mesh can be considered contextual data associated with a three-dimensional data space. [Loughlin & Hughes, 94] suggests annotations imbedded into the data space as a way to help scientists with the storage and retrieval of contextual data. This technique may also be helpful in capturing a scientist's observations.

Several of these suggestions came directly from the observation sessions; others came from matching new technologies with the decomposition of actions identified in Chapter

3. Future development of a three-dimensional rezoner will continue to use these participatory observation methods and rapid prototyping techniques to ensure that the final product supports the work of solving nonlinear three-dimensional finite element problems.

Glossary

affinity diagram: a tool that provides a method for building an overall hierarchical structure of ideas from a set of unstructured ideas. Ideas, issues, general statements from Contextual Inquiry observation sessions can be hierarchically structured using a bottom up process of collecting similar ideas [Cohen, 88].

apprentice-as-designer interview: an observation session used by [Springmeyer, 92] to study scientists as they analyze their data. The observer assumes the role of an apprentice while the expert proceeds with his/her usual activities. Similar to work-based interviews.

artifact walkthrough: a data gathering technique used in Contextual Inquiry when the work occurs over a period of time and/or involves several people. The users and interviewers reconstruct the work process with artifacts by walking through the work process first at a high level and then at lower level reviewing each item.

aspect ratio: A mesh quality measure that defines the squareness of an element. Elements with equal edge length have an aspect ratio of 1.0. Rectangular elements with the height of the element half as long as the width of the rectangle have an aspect ratio of 0.5.

data flow diagram: "a network representation of a system. The system may be automated, manual, or mixed. The Data Flow Diagram portrays the system in terms of its component pieces, with all interfaces among the components indicated" [De Marco, 78] (see Figure G-1).

Figure G-1. Data Flow Diagram [De Marco, 78].

future scenario: a data gathering technique used in Contextual Inquiry when new technology is being applied to an existing work process and/or the user is to be involved in the design process. An artifact walkthrough is conducted first. New technology is reviewed prior to brainstorming about the use of new technology in the work place.

information model: a graphical and textual description of the real world in terms of objects, attributes, and relationships. Objects are abstractions of similar things or instances. Attributes are characteristics of these instances, and relationships are the associations between objects. Information models are represented by Information Structure Diagrams (see Figure G-2) that declare objects, attributes, and relationships; and Object and Relationship Specification Documents that articulate the meaning of objects, attributes, and relationships.

Figure G-2. Example of Information Structure Diagram [Shlaer & Mellor, 88].

linearity: a relation between two quantities (for example, stress and strain) when a change in one produces a directly proportional change in the other [Goudreau, 93].

mesh: an array of grid points labeled and interconnecting in some way to define a discretization of the domain as a union of mesh cells or elements.

nonlinearity: means the proportionality between two quantities breaks down, as it does when a car bumper hits a barrier [Goudreau, 93].

post-observation inquiry: a data gathering technique used in Contextual Inquiry when the work being observed cannot be interrupted. The interviewer observes the user performing some time critical task. Immediately after the user has completed the task the interviewer reviews his/her notes with the user.

prototype/test drive: a data gathering technique used in Contextual Inquiry to review system designs to see how they should be altered to match the work. Start by asking the user to talk about the work, what works well, and what does not work very well. Show

the user the prototype tool and ask for his/her reactions. This differs from a usability study in that the interview is conducted in the work place and a dialog exists between the user and the interviewer.

redistribution: with fixed number of grid points allowing the grid points to move to equidistribute the error by clustering in regions where more finely graded mesh is warranted. Redistribution can also be referred to as grid optimization or rezoning.

refinement: addition of new mesh nodes into an area of interest.

rezoning: same as redistribution.

skewness: A mesh quality measure measuring the orthogonality of an element. If the element is orthogonal, then all edges are perpendicular and the skewness measure is 0. If the skewness measure is 1.0, then at least one pair of angles are tangent at the node.

work-based interview: a data gathering technique used in Contextual Inquiry where the interviewer observes the user at work and can interrupt at any time to ask questions. Similar to apprentice-as-designer interviews.

work models: "simple drawing techniques that describe the most basic aspects of work relevant to design: the culture and physical environment it takes place in, the pattern of human interaction and communication, and the basic tasks and steps involved in getting the work done" [Holtzblatt & Beyer, 95]. There are four work models to represent each of these aspects: context, physical, flow, and sequence models.

References

- Bodker, Suzanne. "A Human Activity Approach to User Interfaces", *Human-Computer Interaction*, 1989; Vol. 4: pp 171-195.
- Carey, Graham F. *Grid Generation, Refinement, and Redistribution*. Unpublished manuscript, 1992.
- Cohen, Louis. "Quality Function Deployment: An Application Perspective from Digital Equipment Corporation." *National Productivity Review*, Summer 1988; Vol. 7, No. 3: pp 197-208.
- De Marco, T. *Structured Analysis and Systems Specification*. New York: Yourdon Press; December 1978.
- Dovey, Donald and Michael D. Loomis. "GRIZ: Visualization of Finite Element Analysis Results on Unstructured Grids." *Engineering Research, Development and Technology FY 93* (UCRL 53868-93). Livermore, CA: Lawrence Livermore National Laboratory; May 1994: pp. 2-45 to 2-47.
- Dovey, Donald and Thomas E. Spelce. *GRIZ Finite Element Analysis Results Visualization for Unstructured Grids User Manual*. Livermore, CA: Lawrence Livermore National Laboratory (UCRL-MA-115696); October 1993.
- Dukowicz, John K. and Nely T. Padial. *REMAP3D: A Conservative Three-Dimensional Remapping Code*. Los Alamos, NM: Los Alamos National Laboratory (LA-12136-MS); 1991.
- Duncan, Anne S. and Minette A. Beabes. "Contextual Inquiry: Grounding Your Design in Users' Work." *CHI'95 Tutorial*. Denver, CO; 7 May 1995.
- Eiseman, Peter R. "A Control Point Form of Algebraic Grid Generation." *International Journal for Numerical Methods in Fluid*, 1988; Vol. 8: pp 1165.
- Eiseman, Peter R. "Grid Generation for Fluid Mechanics Computations." *Annual Review of Fluid Mechanics*, 1985; Vol. 17: pp 487.
- Eiseman, Peter R. and Zhu Wang. *GRIDMAN: A Grid Manipulation System*. White Plains, NY: Program Development Corporation Research Report (N93-24410); 1992.

Engelmann, Bruce. *NIKE2D A Nonlinear, Implicit, Two-Dimensional Finite Element Code for Solid Mechanics - User Manual*. Livermore, CA: Lawrence Livermore National Laboratory (UCRL-MA-105413); April 1991.

Ericsson, K. A. and H. A. Simon. *Protocol Analysis: Verbal Reports as Data*, Cambridge, MA: MIT Press; 1993.

Feiner, Steven K. and Doree Duncan Seligmann. "Cutaways and Ghosting: Satisfying Visibility Constraints in Dynamic 3D Illustrations." *The Visual Computer*, 1992; Vol. 8: pp 292-302.

Gattass, Marcelo and John F. Abel. "Interactive-adaptive, Large Displacement Analysis with Real-time Computer Graphics." *Computers and Structures*, 1983; Vol. 16, No. 1-4: pp 141-152.

Goudreau, Gerald. "Computational Mechanics at LLNL." *Energy & Technology Review*, September-October 1994: pp. 1-5.

Holtzblatt, Karen and Hugh R. Beyer. "Apprenticing with the Customer." *Communications of the ACM*, May 1995; Vol. 38, No. 5: pp. 45-52.

Holtzblatt, Karen and Beyer, Hugh R. "Contextual Design: Using Customer Work Models to Drive System Design." *CHI 95 Tutorial*. Denver, CO; 7 May 1995.

Holtzblatt, Karen and Hugh R. Beyer. "Making Customer-Centered Design Work for Teams." *Communications of the ACM*, October 1993: pp. 93-103.

Holtzblatt, Karen and Hugh R. Beyer. "Requirements Gathering: The Human Factor." *Communications of the ACM*, May 1995; Vol. 38 No. 5: pp. 31-32.

Keller, Peter R. and Mary Keller. *Visual Clues Practical Data Visualization*. Los Alamitos, CA: IEEE Computer Society Press, 1993.

Kerlick, G. David and Elizabeth Kirby. "Towards Interactive Steering, Visualization and Animation of Unsteady Finite Element Simulations." *Proc. of Visualization '93*. Los Alamitos, CA: IEEE Computer Society, 1993: pp 374-377.

Loughlin, Maria M. and John F. Hughes. "An Annotation System for 3D Fluid Flow Visualization." *Proc. of Visualization 93*. Los Alamitos, CA: IEEE Computer Society, 1993: pp 273-284.

Lu, Ning and Peter R. Eiseman. "A Grid Quality Manipulation System." *Numerical Grid Generation in Computational Fluid Dynamics and Related Field*. Proceedings of the 3rd International Conference of Numerical Grid Generation in Computational Fluid

Dynamics and Related Fields. 3-7 June 1991. Ed. A.S. Archilla, J. Hauser, P.R. Eiseman, and J.F. Thompson. North-Holland: Elsevier Science Publishers, 1991: pp. 607-616.

Maker, Bradley N. "Rigid-Body Mechanics for Finite Element Analysis." *Engineering Research, Development and Technology FY 93* (UCRL 53868-93). Livermore, CA: Lawrence Livermore National Laboratory; May 1994: pp. 2-21 to 2-24.

Maker, Bradley N., Robert M. Ferencz, and John O. Hallquist. *NIKE3D A Nonlinear, Implicit, Three-Dimensional Finite Element Code for Solid and Structural Mechanics User's Manual*. Livermore, CA: Lawrence Livermore National Laboratory (UCRL-MA-105268); January 1991.

Minneman, Scott L. *The Social Construction of A Technical Reality: Empirical Studies of Group Engineering Design Practices*. Diss. Stanford University and Xerox PARC Technical Report (SSL-91-22). December 1991.

Mississippi State University's Engineering Research Center. *NGP User's Guide*. Mississippi State University, n.d.

Shapiro, A. B. *TOPAZ3D - A Three-Dimensional Finite Element Heat Transfer Code*. Livermore, CA: Lawrence Livermore National Laboratory (Rept UCID-20484); 1985.

Shlaer, Sally and Stephen Mellor. *Object-Oriented Systems Analysis: Modeling the World in Data*. Englewood Cliffs, NJ: Yourdon Press, Prentice Hall, Inc.; 1988.

Springmeyer, Rebecca R. "Applying Observations of Work Activity in Designing Prototype Data Analysis Tools." Proceedings of IEEE Visualization 93, October 1993: pp. 228-235.

Springmeyer, Rebecca, Meera Blattner, Nelson Max. "A Characterization of the Scientific Data Analysis Process." Proceedings of IEEE Visualization 92, October 1992: pp. 235-242.

Springmeyer, Rebecca R. *Designing for Scientific Data Analysis: From Practice to Prototype*. Diss. University of Davis, CA and Lawrence Livermore National Laboratory (UCRL-LR-111809). September 1992.

Stewart, John E. and Jamshid S. Abolhassani. "A Graphical Interactive Approach to Structured and Unstructured Surface Grid Quality Analysis." AIAA Computational Fluid Dynamics Conference, 11th, Orlando, FL, July 6-9, 1993. Washington, American Institute of Aeronautics and Astronautics, 1993: pp 589-597.

Tang, John C. *Listing, Drawing, and Gesturing in Design: A Study of the Use of Shared Workspace by Design Teams*. Diss. Stanford University and Xerox PARC Technical Report (SSL-89-3). April 1989.

Thalman, Daniel. *Scientific Visualization and Graphics Simulation*. West Suessex, England: John Wiley & Sons; 1990.

Thomas, P.D. and J.F. Middlecoff. "Direct Control of the Grid Point Distribution in Meshes Generated by Elliptic Equations." *ALAA Journal*, June 1980; Vol. 18, No. 6: pp 652-656.

Thompson, Joe F., Z.U.A. Warsi, and C. Wayne Mastin. *Numerical Grid Generation: Foundations and Applications*. New York, New York: North-Holland, 1985.

Warwick, D. P. and C. A. Lininger, *The Sample Survey: Theory and Practice*. New York: McGraw-Hill Book Company; 1975.

Whirley, Robert G. and Bruce E. Engelmann. *DYNA3D: A Nonlinear, Explicit, Three-Dimensional Finite Element Code for Solid and Structural Mechanics User Manual*. Livermore, CA: Lawrence Livermore National Laboratory (UCRL-MA-107254); Rev 1; November 1993.

Wixon, D., K. Holtzblatt, and S. Knox. "Contextual Design: An Emergent View of System Design." *CHI 90 Proceedings*, April 1990: pp. 329-336.

XYZ Scientific Application Inc. *TrueGrid Manual Version 0.99*. XYZ Scientific Applications, Inc. September 1993.

Appendix A

Interview Questions

Questions asked during pre-interviews, work-based interviews, and artifact walkthroughs.

1. What mesh generators do you use?
2. What solvers do you use?
3. What post-processors do you use?
4. Do the generators or solvers have the capability to smooth or rezone a mesh? If yes, have you used this capability? When do you use it and how often?
5. How do you determine if a mesh is good or good enough? What tools do you use to determine this?
6. What do you do if it isn't good enough?
7. What tools have you used to redistribute nodes in a mesh?
8. Describe the tools you have used.
9. What parts of the tool did you find the most helpful?
10. What parts of the tool did you find the least helpful?
11. How do you select an area to remesh?
12. Given these two meshes, how would you compare them to determine which mesh is better?
13. What measures of success do you use to compare the mesh with the new mesh?
14. How often do you rebuild a mesh? How often do you modify it?

Appendix B

Quality Measure Calculations

```
/*
 * quality.h - Definitions for quality measures.
 *
 *   Lauri A. Dobbs
 *   Lawrence Livermore National Laboratory
 *   September 1995
 *
 * Copyright (c) 1995 Regents of the University of California
 */

#ifndef QUALITY_H
#define QUALITY_H

#include <X11/Intrinsic.h>
/*
 * Quality Measure types.
 */
typedef enum
{
    ASPECT_RATIO,
    SKEWNESS,
} Qlty_type;
/*
 * Boolean operators
 */
typedef enum
{
    AND,
    OR
} Op_types;
extern int aspect_ratio_min;
extern int aspect_ratio_max;
extern int skewness_min;
extern int skewness_max;
extern Op_types vis_op;
extern Qlty_type qlty_exp1;
extern Qlty_type qlty_exp2;
extern Widget measure_opt_1;
#endif
```

```

/*
 * quality.c - Routines for computing mesh qualities.
 *
 *   Lauri A. Dobbs
 *   Lawrence Livermore National Laboratory
 *   September 1995
 *
 * Copyright (c) 1995 Regents of the University of California
 */
#include "viewer.h"
#include "quality.h"
#include "misc.h"
int aspect_ratio_min;
int aspect_ratio_max;
int skewness_min;
int skewness_max;
Qty_type qty_exp1;
Qty_type qty_exp2;
Op_types vis_op;
Widget measure_opt_1;

static void compute_hex_aspect_ratio();
static void compute_hex_skewness();
static float compute_hexnode_skewness();
static void compute_shell_aspect_ratio();
static void compute_shell_skewness();
static float compute_shellnode_skewness();
/*****
 * TAG( compute_share_aspect_ratio )
 *
 * Computes the mesh quality for hex and shell elements.
 */
void
compute_share_aspect_ratio( analy, resultArr )
Analysis *analy;
float *resultArr;
{
    if ( analy->geom_p->bricks != NULL )
    {
        compute_hex_aspect_ratio( analy, resultArr );
    }
    else
        bzero( resultArr, analy->state_p->nodes->cnt * sizeof(float) );
    if ( analy->geom_p->shells != NULL ) compute_shell_aspect_ratio( analy, resultArr );
}

```

```

/*****
* TAG( compute_share_skewness )
*
* Computes the mesh quality for hex and shell elements.
*/
void
compute_share_skewness( analy, resultArr )
Analysis *analy;
float *resultArr;
{
    if ( analy->geom_p->bricks != NULL )
    {
        compute_hex_skewness( analy, resultArr );
    }
    else
        bzero( resultArr, analy->state_p->nodes->cnt * sizeof(float) );

    if ( analy->geom_p->shells != NULL ) compute_shell_skewness( analy, resultArr );
}

/*****
* TAG( compute_hex_aspect_ratio )
*
* Computes the aspect ratio for a given hex.
*/
static void
compute_hex_aspect_ratio( analy, resultArr )
Analysis *analy;
float *resultArr;
{
    Hex_geom *bricks;
    Nodal *currentGeom;
    float d[3][12];
    int b, i;
    int nd[8];
    float edge_len[12];
    float i_average, j_average, k_average;
    float aspect_ij, aspect_jk, aspect_ik;
    float aspect_ratio;
    float *resultElem;      /* Array for the element data. */

    bricks = analy->geom_p->bricks;
    currentGeom = analy->state_p->nodes;

```



```

resultElem = analy->hex_result;

for (b = 0; b < bricks->cnt; b++)
{
    for (i = 0; i < 8; i++)
    {
        nd[i] = bricks->nodes[i][b];
    }

    for (i = 0; i < 3; i++)
    {

        d[i][0] = currentGeom->xyz[i][nd[0]] - currentGeom->xyz[i][nd[4]];
        d[i][1] = currentGeom->xyz[i][nd[1]] - currentGeom->xyz[i][nd[5]];
        d[i][2] = currentGeom->xyz[i][nd[2]] - currentGeom->xyz[i][nd[6]];
        d[i][3] = currentGeom->xyz[i][nd[3]] - currentGeom->xyz[i][nd[7]];

        d[i][4] = currentGeom->xyz[i][nd[0]] - currentGeom->xyz[i][nd[1]];
        d[i][5] = currentGeom->xyz[i][nd[3]] - currentGeom->xyz[i][nd[2]];
        d[i][6] = currentGeom->xyz[i][nd[7]] - currentGeom->xyz[i][nd[6]];
        d[i][7] = currentGeom->xyz[i][nd[4]] - currentGeom->xyz[i][nd[5]];

        d[i][8] = currentGeom->xyz[i][nd[0]] - currentGeom->xyz[i][nd[3]];
        d[i][9] = currentGeom->xyz[i][nd[4]] - currentGeom->xyz[i][nd[7]];
        d[i][10] = currentGeom->xyz[i][nd[5]] - currentGeom->xyz[i][nd[6]];
        d[i][11] = currentGeom->xyz[i][nd[1]] - currentGeom->xyz[i][nd[2]];

    }

    for (i = 0; i < 12; i++)
    {
        edge_len[i] = sqrt(d[0][i]*d[0][i] + d[1][i]*d[1][i] + d[2][i]*d[2][i]);
    }

    i_average = (edge_len[4] + edge_len[5] + edge_len[6] + edge_len[7]) / 4.0;
    j_average = (edge_len[8] + edge_len[9] + edge_len[10] + edge_len[11]) / 4.0;
    k_average = (edge_len[0] + edge_len[1] + edge_len[2] + edge_len[3]) / 4.0;

    aspect_ij = MIN(i_average/j_average, j_average/i_average);
    aspect_jk = MIN(j_average/k_average, k_average/j_average);
    aspect_ik = MIN(i_average/k_average, k_average/i_average);

    aspect_ratio = MIN(aspect_ij, aspect_jk);
    aspect_ratio = MIN(aspect_ratio, aspect_ik);
    aspect_ratio = 1.0 - aspect_ratio;

```

```

    resultElem[b] = aspect_ratio;
}

hex_to_nodal( resultElem, resultArr, analy );
}

/*****
* TAG( compute_hex_skewness )
*
* Computes the skewness for a given hex.
*/
static void
compute_hex_skewness( analy, resultArr )
Analysis *analy;
float *resultArr;
{

    Hex_geom *bricks;
    Nodal *currentGeom;
    float d[3][12];
    int b, i;
    int nd[8];
    float cell_skewness;
    float edge_len[12];
    float skewness;
    float *resultElem;      /* Array for the element data. */

    bricks = analy->geom_p->bricks;
    currentGeom = analy->state_p->nodes;

    resultElem = analy->hex_result;

    for (b = 0; b < bricks->cnt; b++)
    {
        for (i = 0; i < 8; i++)
        {
            nd[i] = bricks->nodes[i][b];
        }

        /* Loop over x, y, and z to find the distance between each node and
        * its neighbor.
        */

        for (i = 0; i < 3; i++)

```

```

{

d[i][0] = currentGeom->xyz[i][nd[0]] - currentGeom->xyz[i][nd[4]];
d[i][1] = currentGeom->xyz[i][nd[1]] - currentGeom->xyz[i][nd[5]];
d[i][2] = currentGeom->xyz[i][nd[2]] - currentGeom->xyz[i][nd[6]];
d[i][3] = currentGeom->xyz[i][nd[3]] - currentGeom->xyz[i][nd[7]];

d[i][4] = currentGeom->xyz[i][nd[0]] - currentGeom->xyz[i][nd[1]];
d[i][5] = currentGeom->xyz[i][nd[3]] - currentGeom->xyz[i][nd[2]];
d[i][6] = currentGeom->xyz[i][nd[7]] - currentGeom->xyz[i][nd[6]];
d[i][7] = currentGeom->xyz[i][nd[4]] - currentGeom->xyz[i][nd[5]];

d[i][8] = currentGeom->xyz[i][nd[0]] - currentGeom->xyz[i][nd[3]];
d[i][9] = currentGeom->xyz[i][nd[4]] - currentGeom->xyz[i][nd[7]];
d[i][10] = currentGeom->xyz[i][nd[5]] - currentGeom->xyz[i][nd[6]];
d[i][11] = currentGeom->xyz[i][nd[1]] - currentGeom->xyz[i][nd[2]];

}

/* Find the length of each edge */

for (i = 0; i < 12; i++)
{
    edge_len[i] = sqrt(d[0][i]*d[0][i] + d[1][i]*d[1][i] + d[2][i]*d[2][i]);
}

cell_skewness = 180.0;

/* Determine the minimum angle between the edges at node 0 */

skewness = compute_hexnode_skewness (0, 4, 8, d, edge_len);
cell_skewness = MIN(cell_skewness, skewness);

/* Determine the minimum angle between the edges at node 1 */

skewness = compute_hexnode_skewness (1, 4, 11, d, edge_len);
cell_skewness = MIN(cell_skewness, skewness);

/* Determine the minimum angle between the edges at node 2 */

skewness = compute_hexnode_skewness (2, 5, 11, d, edge_len);
cell_skewness = MIN(cell_skewness, skewness);

/* Determine the minimum angle between the edges at node 3 */

```

```

    skewness = compute_hexnode_skewness (3, 5, 8, d, edge_len);
    cell_skewness = MIN(cell_skewness, skewness);

    /* Determine the minimum angle between the edges at node 4 */

    skewness = compute_hexnode_skewness (0, 7, 9, d, edge_len);
    cell_skewness = MIN(cell_skewness, skewness);

    /* Determine the minimum angle between the edges at node 5 */

    skewness = compute_hexnode_skewness (1, 7, 10, d, edge_len);
    cell_skewness = MIN(cell_skewness, skewness);

    /* Determine the minimum angle between the edges at node 6 */

    skewness = compute_hexnode_skewness (2, 6, 10, d, edge_len);
    cell_skewness = MIN(cell_skewness, skewness);

    /* Determine the minimum angle between the edges at node 7 */

    skewness = compute_hexnode_skewness (3, 6, 9, d, edge_len);
    cell_skewness = MIN(cell_skewness, skewness);

    resultElem[b] = cos(((double)DEG_TO_RAD(cell_skewness)));
}

hex_to_nodal( resultElem, resultArr, analy );
}

/*****
* TAG( compute_hexnode_skewness )
*
* Computes the skewness for a given node.
*/
static float
compute_hexnode_skewness( e1, e2, e3, dist, edge_len)
int e1, e2, e3;
float dist[3][12];
float *edge_len;
{
    float g12, g13, g23;
    float cos_angle;
    float angle;
    float skewness;
    float skew_angle1, skew_angle2, skew_angle3;

```

```

/* Determine the minimum angle between the three edges coincident to
 * a node.
 */

g12 = dist[0][e1]*dist[0][e2] + dist[1][e1]*dist[1][e2] + dist[2][e1]*dist[2][e2];
g13 = dist[0][e1]*dist[0][e3] + dist[1][e1]*dist[1][e3] + dist[2][e1]*dist[2][e3];
g23 = dist[0][e2]*dist[0][e3] + dist[1][e2]*dist[1][e3] + dist[2][e2]*dist[2][e3];

cos_angle = fabs((double)g12) / (edge_len[e1]*edge_len[e2]);
angle = acos((double)cos_angle);
angle = RAD_TO_DEG(angle);
skew_angle1 = MIN(180.0 - angle, angle);

cos_angle = fabs((double)g13) / (edge_len[e1]*edge_len[e3]);
angle = acos((double)cos_angle);
angle = RAD_TO_DEG(angle);
skew_angle2 = MIN(180.0 - angle, angle);

cos_angle = fabs((double)g23) / (edge_len[e2]*edge_len[e3]);
angle = acos((double)cos_angle);
angle = RAD_TO_DEG(angle);
skew_angle3 = MIN(180.0 - angle, angle);

skewness = MIN(skew_angle1, skew_angle2);
skewness = MIN(skewness, skew_angle3);

return skewness;
}

```

```

/*****
* TAG( compute_shell_aspect_ratio )
*
* Computes the aspect ratio for a given shell.
*/
static void
compute_shell_aspect_ratio( analy, resultArr )
Analysis *analy;
float *resultArr;
{

    Shell_geom *shells;
    Nodal *currentGeom;
    float d[3][4];
    int s, i;
    int nd[4];
    float edge_len[4];
    float i_average, j_average;
    float aspect_ratio;
    float *resultElem;

    shells = analy->geom_p->shells;
    currentGeom = analy->state_p->nodes;

    resultElem = analy->shell_result;

    for (s = 0; s < shells->cnt; s++)
    {
        for (i = 0; i < 4; i++)
        {
            nd[i] = shells->nodes[i][s];
        }

        for (i = 0; i < 3; i++)
        {
            d[i][0] = currentGeom->xyz[i][nd[0]] - currentGeom->xyz[i][nd[1]];
            d[i][1] = currentGeom->xyz[i][nd[1]] - currentGeom->xyz[i][nd[2]];
            d[i][2] = currentGeom->xyz[i][nd[2]] - currentGeom->xyz[i][nd[3]];
            d[i][3] = currentGeom->xyz[i][nd[3]] - currentGeom->xyz[i][nd[0]];
        }

        for (i = 0; i < 4; i++)
        {
            edge_len[i] = sqrt(d[0][i]*d[0][i] + d[1][i]*d[1][i] + d[2][i]*d[2][i]);
        }
    }
}

```

```

    }

    i_average = (edge_len[0] + edge_len[2]) / 2.0;
    j_average = (edge_len[1] + edge_len[3]) / 2.0;

    aspect_ratio = MIN(i_average/j_average, j_average/i_average);
    aspect_ratio = 1.0 - aspect_ratio;

    resultElem[s] = aspect_ratio;
}

shell_to_nodal( resultElem, resultArr, analy, TRUE );
}

/*****
* TAG( compute_shell_skewness )
*
* Computes the skewness for a given shell.
*/
static void
compute_shell_skewness( analy, resultArr )
Analysis *analy;
float *resultArr;
{

    Shell_geom *shells;
    Nodal *currentGeom;
    float d[3][4];
    int i, s;
    int nd[4];
    float cell_skewness;
    float edge_len[4];
    float skewness;
    float *resultElem;

    shells = analy->geom_p->shells;
    currentGeom = analy->state_p->nodes;

    resultElem = analy->shell_result;

    for (s = 0; s < shells->cnt; s++)
    {
        for (i = 0; i < 4; i++)
        {
            nd[i] = shells->nodes[i][s];

```

```

}

/* Loop over x, y, and z to find the distance between each node and
 * its neighbor.
 */

for (i = 0; i < 3; i++)
{
    d[i][0] = currentGeom->xyz[i][nd[0]] - currentGeom->xyz[i][nd[1]];
    d[i][1] = currentGeom->xyz[i][nd[1]] - currentGeom->xyz[i][nd[2]];
    d[i][2] = currentGeom->xyz[i][nd[2]] - currentGeom->xyz[i][nd[3]];
    d[i][3] = currentGeom->xyz[i][nd[3]] - currentGeom->xyz[i][nd[0]];
}

/* Find the length of each edge */

for (i = 0; i < 4; i++)
{
    edge_len[i] = sqrt(d[0][i]*d[0][i] + d[1][i]*d[1][i] + d[2][i]*d[2][i]);
}

cell_skewness = 180.0;

/* Determine the minimum angle between the edges at node 0 */

skewness = compute_shellnode_skewness (0, 1, d, edge_len);
cell_skewness = MIN(cell_skewness, skewness);

/* Determine the minimum angle between the edges at node 1 */

skewness = compute_shellnode_skewness (1, 2, d, edge_len);
cell_skewness = MIN(cell_skewness, skewness);

/* Determine the minimum angle between the edges at node 2 */

skewness = compute_shellnode_skewness (2, 3, d, edge_len);
cell_skewness = MIN(cell_skewness, skewness);

/* Determine the minimum angle between the edges at node 3 */

skewness = compute_shellnode_skewness (3, 0, d, edge_len);
cell_skewness = MIN(cell_skewness, skewness);

```



```

    resultElem[s] = cos((double)DEG_TO_RAD(cell_skewness));
}

shell_to_nodal( resultElem, resultArr, analy, TRUE );
}

/*****
* TAG( compute_shellnode_skewness )
*
* Computes the skewness for a given node belonging to a shell.
*/
static float compute_shellnode_skewness( e1, e2, dist, edge_len)
int e1, e2;
float dist[3][4], *edge_len;
{
    float g12;
    float cos_angle;
    float angle;
    float skewness;
    g12 = dist[0][e1]*dist[0][e2] + dist[1][e1]*dist[1][e2] + dist[2][e1]*dist[2][e2];
    angle = fabs((double)g12) / (edge_len[e1]*edge_len[e2]);
    angle = acos((double)cos_angle);
    angle = RAD_TO_DEG(angle);
    skewness = MIN(180.0 - angle, angle); /* Get angle between 0 and 90 */

    return skewness;
}

```

Appendix C

Command Summary

Two new options have been added to the shared results command to assist with visualizing mesh quality. These new results can be viewed using the pull down **Results** menu or using the **show** command as described below. It is recommended that color interpreting be turned off when viewing these results; use **switch noterp** to turn off color interpreting.

show aspect ratio

The average length of the edges in the i, j, and k directions are computed. The smallest of the six possible ratios is determined (i/j, j/i, i/k, k/i, j/k, k/j) and then this value is subtracted from 1.0. A value of 0 indicates that the edge lengths in all three directions are the same. A value of 0.9 indicates that the smallest average edge length is one-tenth of the largest average edge length.

show skewness

The smallest angle is determined by finding the three angles at each node. The measure of skewness is the absolute value of the cosine of the smallest angle. If the grid is orthogonal, then all edges are perpendicular and the skewness measure is 0. If the skewness measure is 1.0, then at least one pair of angles are tangent at the node.

The concept of peeling has been added to aid in the systematic examination of a mesh. The following commands have been provided to define the peels, peel off layers of the mesh, and to turn off peeling. These commands were developed to be used with the shared results aspect ratio and skewness but could be used with other results.

peel x n

Using the models limit box, start from the minimum x value defining *n* peels ending with the maximum x value. The cut plane is perpendicular to the x axis.

peel y n

Using the models limit box, start from the minimum y value defining *n* peels ending with

the maximum y value. The cut plane is perpendicular to the y axis.

peel z n

Using the models limit box, start from the minimum z value defining n peels ending with the maximum z value. The cut plane is perpendicular to the z axis.

peel $x_f y_f z_f x_t y_t z_t n_x n_y n_z n$

The initial cut plane is defined by the point $x_f y_f z_f$ on the plane and the vector normal $(n_x n_y n_z)$ to the plane. The ending cut plane is defined by the point $x_t y_t z_t$ on the plane and the same vector normal.

on/off rough

Turns on (or off) the rough cutting plane display.

peel n n

Peel n more layers.

peel p n

Replace the last n layers (reverse peeling).

peel off

Delete the cutting planes defined for peeling and return display to original condition.

A new menu has been added to define the quality queries. The results of these queries can then be passed to a rezoner for manipulation of the regions defined by the query. See the **Rezone** menu. This menu will work in conjunction with the picking commands.